Master CCI
Langage machine
Contrôle continu écrit 2009

Durée 1h30 heure, document autorisés, calculatrices et ordinateurs interdits

Prévoir environ 1 heure pour la première partie et 30 mn pour la deuxième partie.

1 Base 2 et conversion asm → C

Les exercices classiques consistent à partir d'un programme C normal, à le convertir en "C expansé" dans lequel
– toutes les variables temporaires nécessaires sont déclarées et
– toutes les constructions algorithmiques sont converties en **goto** et **if ... goto** puis à le traduire en langage d'assemblage.

Il s'agit ici d'effectuer le travail inverse : remonter du langage d'assemblage à du "C expansé", puis de ce dernier à un programme C classique.

On considère le fragment de code en langage d'assemblage suivant.

```
        .text                       .data
        .global calcul      x:      .word   5
calcul:                     format: .asciz   "valabs(x) = %d\012"
debut:  ldr r2,= x
        ldr r0,[r2]                 .bss
        cmp r0,#0           res:    .skip   4
        bge Etiq2

Etiq1:  mvn r1,r0
        add r1,r1,#1
        bal Etiq3

Etiq2:  mov r1, r0

Etiq3:  ldr r3,= res
        str r1,[r3]
        mov  pc,lr
```

Mvn (move not) calcule le complément à 1 de son opérande ($r1_i = \overline{r0_i} = 1 - r0_i$).

En langage C, l'opérateur de complément à 1 est noté **~**.

L'instruction **mov pc,lr** est un branchement de retour à la procédure qui a appelé **calcul**.

Question a : On considère une machine dans laquelle les entiers sont représentés sur 6 bits. Quels entiers naturels et relatifs (écrits en base 10) s'écrivent ainsi en hexadécimal :

1. 0x15
2. 0x35
3. 0x2f

Question b : Quel(s) indicateur(s) teste la condition **ge** (dans le branchement conditionnel) ? En déduire de quel type (long int ou unsigned long int) doit être déclarée la variable x dans le programme C.

Question c : On suppose que lors de l'exécution les sections débutent respectivement aux adresses 0x1000 (text), 0x2000 (data) et 0x3000 (bss). Donner le contenu des registres r0 à r3 en fin d'exécution.

Question d : Donner pour chacun des registres (r0, r1, r2 et r3) utilisés la déclaration en C de la variable temporaire correspondante.

Question e : Donner pour chacune des instructions (excepté mov pc,lr) en langage d'assemblage une instruction équivalente en langage C .

Question f : Donner un programme C équivalent ne contenant ni opérateur ~, ni goto. Quel calcul arithmétique réalise ce programme ?

2 Traduction C → asm

Le corps de la procédure main suivante échange les contenus des deux variables a et b. Rappel pour la traduction en langage d'assemblage : en l'absence d'attribut register dans la déclaration en C, la variable doit être stockée en mémoire.

```
#include <stdio.h>

/* variables globales */
/* hors main          */

short int a = 13;

short int *pta = &a;

short int b = 25;

void main (void)
{
    register short temp;
    register short int *ptb;
    printf ("a = %d, b= %d\n",a,b);
    ptb = & b;    /* instruction 1 */
    temp = *ptb;  /* instruction 2 */
    *ptb = *pta;  /* instruction 3 */
    *pta = temp;  /* instruction 4 */
    printf ("a = %d, b= %d\n",a,b);
}
```

Question g : Traduire (dans le même ordre qu'en C) les déclarations de a, b et pta.

Question h : Traduire individuellement en langage d'assemblage les instructions 1 2, 3 et 4 de main. Votre traduction des trois dernières instructions doit résister à une permutation des contenus des pointeurs (short int *pta = &b suivi de ptb = &a).